

# Goodbye, special properties?

## The advantages of an XML-based approach for storing data structure in SMW

Yaron Koren

Fall 2011 SMWCon

September 22, 2011

# Data structure in SMW

All of the elements that can be used to display and edit data: properties, templates, forms, filters, queries.

Within category, property and filter pages, "special properties" are currently used – semantic properties that are pre-defined by the code.

# Some special properties used for defining structure

**From Semantic MediaWiki:** “Has type”, “Allows value”

**From Semantic Forms:** “Has default form”, “Has alternate form”, etc.

**From Semantic Drilldown:** “Has filter”, “Covers property”, “Has date range”, etc.

**From Halo:** “Has domain and range”, “Has min cardinality”, “Has max cardinality”

# Alternate approach: XML

Have a set piece of XML per "class", defined on a single wiki page, and then either:

- use the XML to generate the other pages, or
- have the data structure be defined directly by the XML

(For the purpose of this talk, “XML” actually means “any simple structured-data format” - JSON could also be used.)

# Why XML?

There are at least 9 reasons.

# Why XML? Reason #1

**Makes modifying the data structure much easier, because all information for a "class" can be stored in one place.**

Currently, even a simple operation like adding a field can involve modifying a template and form, creating a new property, modifying queries that display the data, and (if Semantic Drilldown is used), creating a new filter and modifying the category page.

# Why XML? Reason #2

## **Makes form-based editing possible!**

As with Semantic Forms, having forms always makes editing easier, and removes the need for guesswork and research.

## Why XML? Reason #3

**Eliminates internationalization issues for admins.**

Users/admins no longer need to either know English or find out the names of special properties in their own language.

## Why XML? Reason #4

### **Allows import of data structure.**

A data structure contained in a format like UML or SQL could be imported into the wiki, if conversion scripts were created from those formats into the wiki's XML format.

# Why XML? Reason #5

**Allows export of data structure.**

Similarly, data can be exported in order to be displayed as UML, or even (in the unlikely case) to be moved from the wiki into another application.

# Why XML? Reason #6

## **Allows inheritance.**

"Classes" that are similar to one another could be defined to inherit fields from one another, eliminating redundancy.

Example: "Nightclub" inherits from "Place of business". It inherits fields like "Address" and "Opening/Closing hours" and adds the field "Type of music".

# Why XML? Reason #7

## **Could support internationalization of display.**

Different users could see different text on pages and forms, depending on their language preference, if the XML supported it.

This could apply to both display labels as well as actual content, for the case of enumerated values.

Example: a dropdown with “Small, Medium, Large” is displayed as “Kleine, ..., Grosse”.

## Why XML? Reason #8

**Can result in nicer-looking templates and forms.**

Currently, even simple touches like having alternating colors for rows, or having an explanatory row for each field, can take a lot of work to create and maintain. Automatically-generated forms and templates could take care of that... automatically.

# Why XML? Reason #9

**Can result in better HTML for forms.**

This is ideal HTML for a form input:

```
<label for="author">Author:</label>  
<input type="text" name="author"  
value="" />
```

The `<label>` tag allows for better display across browsers and devices, and allows for better audio navigation by blind people.

## Why XML? Reason #9 (continued)

`<label>` is what's known as "semantic HTML", since it conveys the meaning of its contents.

The irony: the HTML generated by Semantic Forms is not "semantic"!

If forms were defined in some way other than via wiki-text, it would be easier to display proper HTML.

Along with reasons to use XML, there are also potential objections to it.

Let's go through three of them...

# Potential objection #1

**XML doesn't belong on wiki pages!**

*My response:*

You can get used to it.

It's not that much weirder or more technical than the wiki-text table syntax... or form syntax, for that matter. And you don't have to edit it directly.

## Potential objection #2

**What about elements that can belong to more than one "class", like properties or templates? Redundant definitions can lead to problems.**

*My response:*

Hopefully inheritance can avoid having more than one class defining the same property or template, in most cases.

## Potential objection #3

**Can XML really support all the many customizations people can do to templates and forms?**

*My response:*

Probably not. But if this setup can support what people do with templates and forms 80% of the time, it will be worthwhile.

This XML thing sounds pretty good. So  
what's the next step?

...it's already possible! (Sort of.)

# The "Page Schemas" extension

Formerly referred to (while it was still in the planning stage) as "Semantic Classes", and then "Semantic Schemas".

First version was created this summer by Ankit Garg, a college student in India, as part of the Google Summer of Code (I was the project mentor).

Version 0.1 was released in August; currently still in "alpha" stage.

Let's do a demo!